

~ Conference in Montreal (Recon, June 2005) ~

(Date: Saturday 18 June 2005; Location: [Crowne Plaza Montreal Centre](#) downtown Montreal)

Wizard
searching:
reversing the
commercial
web for fun
and knowledge



Back to [searchlores](#)



"Wizard searching: reversing the commercial web for fun and knowledge"

&

"The importance of evaluating your findings"

[Fravia's talk at the Recon](#)

by Fravia+, June 2005, version **3.9**

Approximate
"Scaletta"
(just a "pot pourrie"
of many possible
paths, the talk itself
will probably differ)

[Introduction: arrows,
sword, shield
Proxomitron &
Rapidshare Books
Must know,
Anonymity & whois
Some magic tricks
How deep is deep?
Fravia's copyrighted,
trademarked and
patented anti-EULA
definitive solution
Music & the
streaming scam
10 years from now](#)

[Languages
Eventual sidepaths
Conclusions
Searching for
disappeared sites
Netcraft Structure
of the web](#)

All the main s.e.: [Bk:
flange of myth
\[rose\] webbits'](#)

[cosmic power](#)

This file dwells at
[http://www.
 searchlores.org/
 montreal_2005.htm](http://www.searchlores.org/montreal_2005.htm)
 &
 For best results use a
[Opera](#) browser on a
 17" screen

INTRODUCTION

Arrows (sharp and blunt), Sword & Shield and Structure of the Web

Excuse my English, please, which in fact is not even my first foreign language, mais je craigne que mon francais soit ancor pire, hence I'll stick to english. Please also note that I'm not always going to be politically correct, a severe handicap of mine. For instance I understand "copyright" as just "the right to copy" as much as I fancy. In fact some of the things that we'll see together today could be misinterpreted as 'malicious', 'inconvenient' or even 'slightly illegal' in some "copyright obsessed" countries. But finding out and explaining how things work is one of the many tasks of the reverser, as everyone here knows. And we must fullfill it, even in face of adversities and persecution :-)

Another defect of mine is that I find "powerpoint style" slides during talks pretty disgusting: I use htm code instead. I'm sure you'll be deceived, but at least you can follow this talk on your own screens using http://www.searchlores.org/montreal_2005.htm.

With this talk I would like to give you some [cosmic power](#), no more and no less.

Of course in one hour we wont be able to examine the many searching techniques in depth. I would just like to see with you some paths that are useful to search effectively the web. Just build on your own onto what you'll hear today and you wont need nooune nomore to explain *you* anything: you'll always quickly fetch your own signals among the heavy commercial noise that infests the web.

Please try to understand [the reasons behind](#) the querystrings we will see together, don't just count on the querystrings themselves: the specific arrows we will launch today will not remain "sharp" for long: everytime I use a querystring to make a point during a lecture, that same querystring is often immediately re-used many times over a very short time span, thus 'affecting' the web.

A sort of [schroedinger's cat](#) effect: like arrows, our new sharp querystrings -once used- slowly become blunt. Do not worry: your skill and understanding of the web will allow you to produce new, sharper ones. As many as you want.

Note also that most examples -though valid no matter which operating system you'll use- will apply here to [windoze](#), that I am using right now.

While I believe that GNU-Linux is much better (**not** "Linux": "Linux" does not exist. The name is [GNU-Linux](#)), I still prefer to use windoze as a matter of "cracking preference": it's waay more fun (and useful) to [reverse programs](#) and operating systems you do not happen to have the source code of :-)

The interesting side of a commercial infested web is that those very databases that have been created in order to sell (or to hoard) files (huge repositories of musics, books, images, software, you name it) lay open , or -ahem- ["next to open"](#), at our disposal, once we learn some basic searching skills. Also: there's not only a 'commercial web' to explore and conquest. The web of knowledge is still alive and kicking, albeit uncomfortably buried underneath the sterile sands of the commercial desert.

In fact seekers possess a "double edge": a seeker knows how to find [the free knowledge](#) that the web still offers

underneath the morasses of frill, AND he knows how to enter the commercial databases and liberate knowledge [that someone does not want to be free](#).

[The web was made for SHARING, not for selling and not for hoarding](#), so -as we will see- its very "building bricks" deny to the commercial vultures the possibility of enslaving parts of it.

Once you know how to search the web and become "a seeker", the entire human knowledge will become available, at your command and disposal, no matter where, or how, somebody may have "hidden" it.

Let's start with a simple example, let's imagine we would use google "like Joe Luser" in order to search some anti-streaming software for free, let's say we want to find "total recorder" (more on anti-streaming software [later](#)):

["index of" warez "total recorder"](#)

Alas, this blunt and broken arrow will have your target signal submerged under such a heavy commercial noise that the results will be next to useless.

Try clicking the resulting links: commercial crap -in this case- mostly by the wankers at <http://search.biz.tm> that have spammed google "à la va banque", tarning their useless services as legit google results (the [tm suffix](#) is a hiding place for all sort of web-spammers. Good search engines should simply delete all tm sites from their indexes).

So the query we just saw is seriously flawed. What kind of arrow should we use instead?

Well, a query like the following should cut more mustard:

[\("wares" OR "warez" OR "appz" OR "gamez" OR "abandoned" OR "pirate" OR "war3z"\) \("download" OR "ftp" OR "index of" OR "cracked" OR "release" OR "full"\) \("nfo" OR "rar" OR "zip" OR "ace"\) +"total recorder"](#)

Btw, note how useful such a query is even WITHOUT the specification [+"total recorder"](#).

This is just a simple example, and using just one of the [main](#) search engines. Always remember "*google alone and you'll never be done*": there are many good and powerful search engines: at the moment the most important ones are [google](#), yahoo (with its [philttron](#) slider), msn (with its wondrous [sliders](#)), and [teoma](#).

Btw, I hope you know WHY there are so many search engines. Basically for the same reason that my site is apparently quite messy and labyrinthical... 🚫

Just repeating a given query with different engines will give you slightly different results, notwithstanding their overlapping and the fact that altogether the search engines cover -at best- just slightly more than [just a third](#) of the whole web... 🚫

Let's first have a short look at what [the web looks like from a searcher's point of view](#).

[Outside linkers](#) are fetched through klebing (and stalking and social engineering), [the bulk](#) and the [outside linked](#) through combing and short and long term seeking, the hidden and commercial databases through [password breaking](#) or [guessing](#), social engineering or, more simply, just seeking one of the many lists of databases' hardcoded passwords (à la [Borland Interbase's "politically correct"](#)) on the web.

Here for instance one of these lists: [defpasslist1.htm](#), here a [better one](#) (and you may also want to delve into some common php scripts [pestering php commercial vultures](#))

To -quickly- fetch his targets, a seeker needs to "wade" through the slimy commercial morasses of the web, made specifically "ad captandum vulgus", and to "cut" all the useless ballast.

The Sword

But when you need to "cut" the Web your arrows, even the best ones, wont be enough: you'll need first of all a SWORD, a sharp blade: [a capable and quick browser](#).

That's the first and foremost tool of a seeker. MSIE, Microsoft Internet explorer is a no-no-no, buggy, bloated and prone to all sort of nasty attacks. The two current "philosophical schools" are either Firefox or [Opera](#)... which is the quick browser I mostly use for a plethora of [reasons](#)... 🚫

The Shield

Whichever browser you use, no sword will be enough without a SHIELD. And your shield, and a mighty one for that, is [proxomitron](#).

Proxomitron is a very powerful tool. Its power lies in its ability to [rewrite webpages on the fly](#), [filter communications between your computer and the web servers of the sites you visit](#), and to [allow easy management of external proxy use](#). Here a link to an old, but very good essay about proxomitron basic installation: [anony 8.htm](#), and a link to another essay, [Uncle Faf goes inside proxomitron](#) about further fine-tuning.... Let's sum it up: "*Only morons 'just do it' without Proxomitron.*"

WHY PROXOMITRON

(The 'rapidshare' example)

The web was made for sharing

Among the many useful uses of proxomitron, its filters offer more speed to the seeker: as an example let's use proxomitron to nullify the time waiting span imposed by [rapidshare](#).

Rapidshare is one of those "upload repositories" where people can upload large files. It allows unlimited downloads. There are many other similar repositories:

[YouSendIt](#): 1 Giga max, after 25 downloads the file is removed

[Sendmefile](#): 30 Mb max, after 14 days the file is removed

[Megaupload](#): 500 Mb max (!), after 30 days unused the file is removed (like rapidshare)

[qfile.de](#): 50 Mb max, after 30 days unused the file is removed (like rapidshare)

and so on...

Rapidshare searches [are worth a small digression per se](#): Let's imagine you are interested in, say "oracle":

[rapidshare.de/files oracle](#) (or, using MSN's sliders: [{frsh=94} {popl=20} {mtch=99} rapidshare.de/files oracle](#))

Such kind of searches will give the seekers aplenty fruits:

http://rapidshare.de/files/1438759/McGraw_Hill_Oracle_Application_Server_10g_Admin_Handbook.rar.html
http://rapidshare.de/files/1438839/McGraw_Hill_Oracle_Database_10g_High_Availability_with_RAC_Flashback___Data_Guard.rar.html
http://rapidshare.de/files/1438861/McGraw-Hill_Osborne_Oracle_Database_10g_SQL.rar.html
http://rapidshare.de/files/1438902/McGrawHill-Oracle_Database_10g_New_Features.rar.html
http://rapidshare.de/files/1438921/Oracle_Database_10G_-_Automatic_Sga_Memory_Management.rar.html
http://rapidshare.de/files/1438930/Oracle_Database_10G_-_Automatic_Storage_Management.rar.html
http://rapidshare.de/files/1438946/Oracle_Database_10g_-_DBA.rar.html
http://rapidshare.de/files/1438968/Oracle_Database_10g_-_New_Features_PPT_.rar.html
http://rapidshare.de/files/1438991/Oracle_Database_10g_-_Proactive_Space___Schema_Object_Management.rar.html
http://rapidshare.de/files/1439016/Oracle_Database_10g_-_SQLAccess_Advisor.rar.html
http://rapidshare.de/files/1439096/Oracle_High_Performance_Tuning_for_9i___10g_digital_press_.rar.html
http://rapidshare.de/files/1439711/Oracle_10g_2Day_Training.rar.html

Alas! Rapidshare, while useful, has a silly commercial attitude with an annoying "delaying" trick.

Let's have a look at it, fetching a book, that could maybe prove of some interest for some of the worthy colleagues that have gathered here today:

<http://rapidshare.de/files/1709371/Wiley.Reversing.Secrets.of.Reverse.Engineering.Apr.2005.eBook-DDU.zip.html>

here is an example of rapidshare 'delaying' javascript code, which runs on client side:

```
<script>var c = 58; fc(); function fc(){
if (c>0){document.getElementById("dl").innerHTML = "Download-Ticket reserved. Please wait " +
c + ' seconds.
Avoid the need for download-tickets by using a PREMIUM-Account. Instant access!';
c = c - 5;setTimeout("fc()", 5000)} else {document.getElementById("dl").innerHTML = unescape('
```

```
%3C%68%32%3E%3C%66%6F%6E%74%20%63%6F%6C%6F%72%3D%22%23%43%43%30%30%30%30%22%3E%20%44%6F%77%6E%
6C%6F%61%64%3A%3C%2F%66%6F%6E
%74%3E%20%3C%61%20%68%72%65%66%3D%22%68%74%74%70%3A%2F%2F%64%6C%31%2E%72%61%70%69%64%73%68%61%
72%65%2E%64%65%2F%66%69%6C%65
%73%2F%31%34%33%38%37%35%39%2F%32%37%37%35%37%30%39%37%2F%4D%63%47%72%61%77%5F%48%69%6C%6C%5F%
4F%72%61%63%6C%65%5F%41%70%70
%6C%69%63%61%74%69%6F%6E%5F%53%65%72%76%65%72%5F%31%30%67%5F%41%64%6D%69%6E%5F%48%61%6E%64%62%
6F%6F%6B%2E%72%61%72%22%3E%4D
%63%47%72%61%77%5F%48%69%6C%6C%5F%4F%72%61%63%6C%65%5F%41%70%70%6C%69%63%61%74%69%6F%6E%5F%53%
65%72%76%65%72%5F%31%30%67%5F
%41%64%6D%69%6E%5F%48%61%6E%64%62%6F%6F%6B%2E%72%61%72%3C%2F%61%3E%3C%2F%68%32%3E' )
}}</script>
```

In this case you would just use following proxo filter (by Loki):

```
Name = "RapidShare"
Active = TRUE
URL = "*rapidshare.de*"
Limit = 256
Match = "(var count?)\1 = [#0:45]"
Replace = "\1 = 0"
```

The other limit of rapidshare, the 'just one download' limit (that I bet some of you have already encountered in the past few minutes :-)) can of course also be circumvented, for instance using rotating anonymous proxies, a task made easy(*) by our good ole PROXOMITRON.

Alternatively you can flush and request a new IP address:

```
Start --> run --> cmd.exe --> ipconfig /flushdns --> ipconfig /release --> ipconfig /renew --> exit
```

Erase your cookies and reconnect to rapidshare.

Morale of the whole story? Shap sword + Powerful shield (+ good arrows) = as many useful books (inter alia) as you wish.

Books

There's a whole section regarding [books searching](#) at [searchlores](#), and you can delve into it by yourself. Suffice to say that (almost) all books mankind has written are already on the web somewhere, and that while we are sitting here dozens of fully scanned [libraries](#) are going on line: if you'r attentive enough, and if your searching scripts are good, you can even hear the clinking "thuds" of those huge databases going on line... 🗡️

For instance, fitting for Quebec: <http://gallica.bnf.fr/scripts/catalog.php?Sujet=%22Quebec%22> which is part of the following search engine: <http://gallica.bnf.fr/> and of its huge database of books, not only in french: <http://gallica.bnf.fr/scripts/catalog.php?Sujet=%22Rhetorique%22>.

There are many huge databases of books on the wide web. Private, public, educational or commercial? It is irrelevant for seekers. In order to fetch your target you just need some correct strings, i.e. -as usual- some sharp arrows.

A banal, yet useful approach is starting from [the powerful A9 engine](#), for instance, for Conan Doyle, <http://a9.com/conan%20doyle?a=oobooks> and then fetch its own [study in scarlet](#).

Of course once we have [some sharper arrows](#), it is relatively easy to fetch whole copies of a given book onto the web at large...

This is true for all kind of copyrighted books as well... let's see: ["Suddenly, caught by the level beams, Frodo saw the old king's head"](#)...

and we land here, for instance: [Suddenly, caught by the level beams, Frodo saw the old king's head: it was lying rolled away by the roadside. `Look, Sam!' he cried, startled into speech. `Look! The king has got a crown again!'](#)

Finally, try out (and understand) this arrow:

[-inurl:htm -inurl:html intitle:"index of" +\("/ebooks"|"book"\) +\(chm|pdf|zip\) +"For Dummies"](#)

or maybe you prefer this one?

[-inurl:htm -inurl:html intitle:"index of" +\("/ebooks"|"book"\) +\(chm|pdf|zip\) +"o'reilly"](#)

(anyway at the moment with books even [banal arrows](#) will deliver whatever you want)

MUST KNOW

sine qua non

Just a short tour around the house:

[Main](#), [regional](#) and [local](#)

[ftp](#), [blogs](#) and [targets](#)

[usenet irc](#) and then, of course, [trols](#)

Again: [anonymity](#) and [stalking](#) (and [luring](#))

The importance of anonymity

This brings us to a very interesting contradiction: on one site "echelon" and the total "big brotherish" control, on the other "[wardriving](#)" and pretty good anonymity... 🚫

A relative guide to anonymity, by fravia+, June 2005

RULES

1. buy laptop cash elsewhere (not with credit cards and not where they know you)
2. wardrive in another part of the town, not the one you live in
3. download only, or if you upload, upload only anonymous things or PGP encrypted stuff
4. rotate your wifi card mac address at every access point: I use "Macmakeup"
5. use wardriving laptop ONLY FOR THAT, no personal data whatsoever on it

TECHNIQUES

1. Find speedy, beefy first wifi accesspoint with netstumbler: there are so many unprotected at all that you don't even need to bother firing a wep-packets-analyzer to crack weak WEP-encryptions
2. connect, browse, download, all shields down, javascript, java, the whole bazaar: who cares?
3. ISP "A" will register everything "he" does.
4. work half an hour, download the helluja out of it, upload with care
5. walk/drive ten meters: change access point
6. ISP "B" will now register everything "another he" does.
7. work half an hour, download the helluja out of it, upload with care
8. walk/drive ten meters: change access point
9. ...repeat at leisure...
10. (reformat hard disk -or restore image- every week or so, just in case)
11. next day another part of the town, or another town :-)
12. and so on...

Examples of "one shot" email addresses..., always useful for "quick contact" purposes

Mailinator <http://www.mailinator.com/mailinator/Welcome.do>

Another example:

<http://www.pookmail.com/>

How to discover whois

For instance using the previous example: <http://www.whois.sc/pookmail.com> (scroll down for contact names and info)

Some Magic

as promised, some assorted "wizard" tricks...

1) Sourceror2 (by Mordred & rai.jack)

[try it right away](#)

Right click and, in [opera](#), select "add link to bookmarks"

```
javascript: z0x=document.createElement('form'); f0z=document.documentElement; z0x.innerHTML = '<textarea rows=10 cols=80>' + f0z.innerHTML + '</textarea><br>'; f0z.insertBefore(z0x, f0z.firstChild); void(0);  
javascript:document.write(document.documentElement.outerHTML.replace(new RegExp("<","g"), "<"));
```

2) Another google approach

<http://www.google.com/complete/search?hl=en&js=tru%20e&qu=fravia>

3) Another google approach (by Mordred)

Here is a way to gather relevant info about your target

["index+of/" "rain.way*****"](#)

Useful to see date [and size](#) that follow your target name...

4) ElKilla bookmarklet (by ritz)

[try it right away](#) (no more clicking, press DEL to delete and ESC to cancel)

Right click and, in [opera](#), select "add link to bookmarks"

More about bookmarklets in the [javascript bookmark tricks](#) essay (*).

How deep is deep?

a taste of the seekers' web: some [images](#) (pr0n & nopr0n)

Let's have a look at the depths of the web...

A search engine for [GREEK PI](#) number sequences... (pointed out by 8~) on a sunny Juny day)

Ok, that's boring... let's see...

Uhhmm. Is this a "mature" audience?

[intitle:index.of +playmates -hot -"free pics" -filetype:htm -filetype:html](#)

Or, if you want something more vulgar,

[\(blnd3 OR blonde3 OR blond3\) intitle:index.of](#)

...the only limit to such arrows is, as always, your own creativity :-)

Besides the difference between pr0n and art is always in the [eye of the beholder](#): [Gustave Courbet The Origin of the World](#).

Hence let's continue with art: ["The betrayal Of Christ" caravaggio](#) for instance, for a quick "caravaggio" search, or [caravaggio33.jpg](#) to make sure they have at least as many images :-)

Finding [images](#) on the web is rather simple. Let's make another small 'caravaggio' search using some [ad hoc images search engines](#):

[cardsharps](#) as you can see, there are [many good versions](#) of this specific masterpiece.

And more: in fact most images search engines will give you [the signal you expect](#).

Now, the interesting point is that we can find OTHER versions of a given target image applying some 'fancy' arrows: [Oil on canvas, 90 x 112 cm Kimbell Art Museum](#)

Remember -however- that on the web you always need to [evaluate](#) what you find!

We are not limited to images: of course we may use similar approaches for music

You like music of, say, the seventies?

[\("seventies2.htm" OR "seventies2.html"\) music](#)

But before going over to music searching let's indulge into a small 'reversing' digression...

Fravia's copyrighted, trademarked and patented anti-EULA definitive solution

Is there a lawyer in the house?

Before beginning the next snippet, about streaming, we would like to show how to operate in a totally legal (sortof) way. [iradio_setup.exe](#) (or maybe here: [iradio_setup.exe](#) : [iradio](#) is a internet radio grabber and ripper, a useful, but alas commercial, program that will allow anybody to intercept and register on the fly any broadcasted mp3. It is therefore *eo ipso* a good anti-streaming tool.

Its protection routines are ludicrous, suffice to say that if you disassemble it you'll find even the following inside its code:

```
"D:\Jobs\3alab\RadioGrab\src\protection\.\ASProtection\lc.h"
```

Where we can see inter alia that the original name was probably "RadioGrab".

but I wont go into its protection routines now, I just wish to demonstrate "how to nuke a EULA", you know those END-USER-AGREEMENT-LICENSES that nobody reads when clicking onto install files, even if -for all you know- they may impose you to sacrifice your first-born to their gods.

Apart prohibiting disassembly, a sin I cannot condone, IRADIO's EULA carries the following surreal mumbo-jumbo:

```
All title and intellectual property  
rights in and to the content that may be accessed
```

through use of this SOFTWARE PRODUCT remains the property of the respective content owner and is protected by applicable copyright or other intellectual property laws and treaties. This EULA grants you no rights to use such content.

Now, we cannot accept this, because, to be frank with you, the very reason we might want to install this anti-streaming grabber on our laptop is to grab music that may happen to be patented :-)

So we not only disagree, we STRONGLY disagree and do not accept this eula.

So let's fire our [customizer](#), and let's "strongly disagree" to this EULA before installing iradio...



From what you saw, follows my [copyrighted, trademarked and patented anti-EULA definitive solution](#):

Either EULAs [ARE](#) legally binding, in which case this "EULA of ours" is legally binding as well, or EULAs [ARE NOT](#) legally binding, hence (as I always thought) they are just pseudo-juristical high-sounding overbloated crap.

Quod erat demonstrandum: EULA owners -all over the world- please choose, we'r happy either way :-)

Music

the streamers rebuked

Speaking first of normal, "non streamed" music, a completely new wave of music searching is due to the relatively recent [mp3 blogs](#) phenomenon. this said, blogs -in general- are mostly so boring (and short-lived) that its mostly a waste of time to visit them: usually it is MUCH simpler to just fetch the music you need from web repositories and databases any time you fancy it.

A simple music searching approach (coz [m4as](#) are less censored than [mp3s](#)):

[m4a "index of" dylan](#) For instance:

<http://www.stud.ntnu.no/~nikgol/21-11-2004/>

And you'll also land inside this huge mp3 pasture, so big that it may crash evena mighty browser (try it with firefox and it will probably grind)...

<http://24.91.184.80/jservlet/files/music/>

A more complex mp3 "klebing" webbit:

["icons/sound2.gif" "index of" mp3](#)

See? Now you'll have to "peel the URL-onion", backwards, towards the correct targets.

The streaming scam

More and more music snippets (and videos) are STREAMED on the web. While there are [very simple](#) ways to defeat any streaming protections, some good anti-streaming tools are a *sine qua non* on anyone's box. Fittingly -I believe- for a recon conference, I will show today how to reverse two such tools: a general connections checker and a streams downloader.

To individuate what exactly is going on during your seeking connections, without tedious studying of your ethereal or

firewall loggings, you may choose to use a small [traffic checker](#) called [ipticker](#): a very useful tool, small, not intrusive, powerful.

Ipticker is useful in order to check connections and TCP/UDP data, hence you can use it to check suspicious activities (for instance when visiting rogue web sites), and [you can use it also to quickly gather the real URL of all streamed files](#). You can download [here](#) the old (and uncracked) version 1.6. of ipticker. [They are at version 1.9](#) now, so I hope they'll be grateful for this kinda 'advertisement' of their (good) software and pardon me the following look under the hood.

This older version of Ipticker was indeed **very badly** protected: A quick grep for "UNREGISTERED VERSION" will land us smack inside the following useless "protection" routine (archaic, I know, but this is a talk for bourgeois, real crackers in the audience should please refrain from laughing). Only four instructions need a comment.

```
:4039BE E833DFFFFFF call 4018F6 ; --> do incredibly complex calculations on the
registration key
:4039C3 85C0 test eax, eax ; --> test result of incredibly complex calculations.
Al=0?
:4039C5 7512 jne 4039D9 ; --> No: jne "good guy"
:4039C7 6824D34000 push 40D324 ; --> Yes: push "UNREGISTERED VERSION"
and flag "bad guy"
```

Should somebody want to be a "good guy" he may just modify the ONE byte in red above, turning that "jump if not equal" into a "jump if equal" ([74](#)) instruction...

In order to "automate" the stream downloading itself, a very useful program is [STREAMDOWN](#), a [streaming media download tool](#). It supports not only HTTP and FTP download, but also most streaming media download protocols, such as RTSP, MMS, MMSU, MMST. It also supports download resuming. You can download Windows Media Streams (.ASF, .ASX, .WAX, .WMA, .WMV), Real Video/Audio Streams (.RM, .RAM, .SMIL) and/or .MP3

It is a useful program to counter those clowns that happily stream music and films in order to avoid people making copies of it (even legitimates copies for personal use), but its registration routine is another classical example of [a completely useless protection scheme](#), hence maybe of some interest for this audience.

Here I present the older [version 3.3](#) of streamdown, [they are now at version 5.0](#), so I hope they'll be grateful for this kinda 'advertisement' of their (good) software and pardon me the following look under the hood.

A quick grep for "regcode" (or for "unregistered version") will land us smack inside this "protection" snippet of the code (archaic stuff nowadays! Don't laugh please, software is still "protected" this way).

I have shortened the code for quicker comprehension:

```
:0040A5C4 68BE625000 push 005062BE <----- (Data Obj ->"RegCode")
... do stuff with & check length of previously entered strings "RegName and RegCode"...
:0040A5D5 E8A62E0200 call 0042D480 <----- StreamDown.NEW_00_KEYCHK_CSD:
mov byte ptr [00507FB5], 01 if legit key
... test return from StreamDown.NEW_00_KEYCHK_CSD
:0040A5DE 0F84E2000000 je 0040A6C6 <----- towards exit with bad flag 00
if bad strings
... do irrelevant stuff...
:0040A601 FF5218 call [edx+18] <----- again: is it a legitimate
code?
:0040A604 84C0 test al, al <----- test result of checking
routine: this time equal (0) if legit
:0040A606 0F8587000000 jne 0040A693 <----- non equal? Horror: jump to
bad guy and avoid registering
:0040A60C 66C745DC0800 mov [ebp-24], 0008 <----- registered legitimate paying
user routine ----|
... do various good stuff for the legitimate registered
user... |
:0040A627 BAC7625000 mov edx, 005062C7 <----- (Data Obj ->"Register to:
") |
... make clear he's registered and
legit... |-- "good guy"
routine
:0040A685 B001 mov al, 01 <----- good guy flag high on the
```

```

pennon |
... prepare edx
register ...
|
:0040A691 EB3F          jmp 0040A6D2          <----- avoid bad guy flagging & go
to good exit -----|
:0040A693 66C745DC1400      mov [ebp-24], 0014    <----- (Jump from
Address :0040A606): start code for the "bad guy"
:0040A699 BAD6625000    mov edx, 005062D6    <----- (Data Obj ->"Unregistered
Version")
... do evil stuff, decrease counter and mark "bad guy" (or bad strings) with bad flag 00 ...
:0040A6D2 5B          pop ebx              <----- (Jump from
Address :0040A691): prepare exit
... pop sequitur and exit
:0040A6D6 C3          ret

```

As anyone can see, a simple 0F8487000000 ([je 0040A693](#)) instead of that 0F8587000000 ([jne 0040A693](#)) will automatically transmute bad guys into good guys (as unlock codes I used my nick and 12 random numbers as key, if I am not mistaken). When will software programmers learn some more useful [protection approaches](#)?

Cracking is anyway useless if you know how to search: it wouldn't for instance be difficult to find a ready made crack for this software (or for anything else). For instance:

[{frsh=89} {popl=21} {mtch=16} crack streamdown](#)

Note that you may even use any ad hoc, non porn infested, [crack search engine](#).

But using ready made cracks is not elegant and hence "deprecated": you should always crack your own software by yourself :-)

Here a streamed example (from yahoo):

<http://playlist.yahoo.com/makeplaylist.dll?>

sid=11321646&segment=0&s=1808403968&ru=y&b=fc7udgd19lhn0429ac805&type=m

And streamdown will give you the microsoft media service URL

<mms://wmcontent44.bcst.yahoo.com/bus01root6/Bus01Share22/ShowBiz Network/2/11321646.wmv>

That's it for streaming. With these two small programs, plus iradio, you'll be able to tackle the most common kinds of stream.

10 years from now.

What's that huge terabyte inside your trousers?

Terabyte, Noun: *A unit of information equal to one trillion (1,000,000,000,000) bytes*

In your pocket, yes, yours, soon enough, 10 terabytes of stuff, on a light, tiny, wonderfully crafted small, shiny gadget. Maybe a coupla "petabytes" (*) later.

Enough to hold [the biggest libraries on earth](#) inside your pocket. Enough to have big chunks of the web inside your pocket.

10 terabytes of stuff... Woah!

But what stuff?

music, books, images, films, software, yes yes, all the things we have seen we can find on the web. A lot of them. So much that you could legitimately wonder if it does matter at all what you'll pack inside.

Should you really care to choose if you can have anything and then some? Just grab and use.

You'r gonna have so much space in that little shiny gadget inside your pocket: just hoard everything and then "choose" later...

HA! Soon or later you'll have to choose.

Soon or later your limited life-span will reclaim your attention and your own phisical rotting will compel you to react to the rotten meals of the "fast knowledge food" chains that have poisoned your brain...

Yep, yes, yes. You already know all this browsing today's web: but you'll encounter it amplified a zillion times in less than 10 years... We **can** have everything, but we **wont** have everything. We have to choose, to operate choices.

Hence the importance of EVALUATION, right now of course, yet even more in the near future.

What is and, foremost, what **will be** very soon very important? Not (only) **to seek and find!** In fact you'll have everything and then some... auto-magically... if you want, you'll even have copy of huge chunks of the web inside your pocket, already in a few years time!

So you'll not need to rely anymore on someone else advice, or some search engine algos, helping you to find whatever you wish through its more or less good approaches. You'll have ALL the content (and ALL the crap) at your own personal disposal... **so the problem will be to discard the crap** (the same problem that we already have today on the web). Back to case one:

the need to know how **TO EVALUATE**, an important lore **already now**, knowing how to evaluate (if possibly quickly) what you find will be **the most important knowledge** in a few years from now.

I think we should maybe move onto "advanced techniques of evaluation": in fact our "advanced searching techniques" are now being explained (and also ripped and sold) by more and more newbyes. Despite their incompetence that's good: it means that the snowball is already rolling :-)

Evaluating will be more and more important: you see: There are still some PHYSICAL constrictions today, in a bookshop, a museum, a library... you cannot, physically just put everything the human race has produced inside it. There's simply not enough space, "physically" speaking, so you have (or at least someone has) to operate CHOICES.

In a few years, on the virtual landscape, such choices may be on one side **UNNECESSARY** (since you can or will very soon have all the cakes and also the possibility to eat all of them) and on the other side **IMPERATIVE** (since physically - again- you as a human being with a limited time span and hence cannot have all the cakes and also eat all of them)

Ah, the wondrous contradictions of the web :-)

So you'll soon have -say- 10 terabytes (or more) in your pockets. 10 terabytes of WHAT?

Learning to discern CRAP and learning to reverse advertisers' tricks will be MORE and MORE important.

Your capacity of not being fooled, of understanding the rhetorical tricks will be PARAMOUNT (even more than now... "scusate se è poco")

Your capacity to choose: in life and on the web.

The correct (useful, lasting) cloth or textile, the correct (tasty, lasting) pear fruit, the correct (enlightening, lasting) book to read, the correct (involving, lasting) game to play. The common word for these phisical things is probably that LASTING adjective. And this will -I believe- be even more valid for the web, with all its terabytes of moving virtual quicksand.

That's it.
Any questions?

SEARCHING FOR DISAPPEARED SITES

<http://webdev.archive.org/> ~ The 'Wayback' machine, explore the Net as it was!

Visit [The 'Wayback' machine](#) at Alexa, or try your luck with the form below.

Alternatively ,learn how to navigate through [\[Google's cache\]](#) (without images if you want a relative anonymity, just add "&strip=1" will give you the text version of google's cache)!

NETCRAFT SITE SEARCH

(<http://www.netcraft.com/> ~ Explore 15,049,382 web sites)

VERY useful: you find a lot of sites based on their own name, which is another possible way to get to your target...

Search:

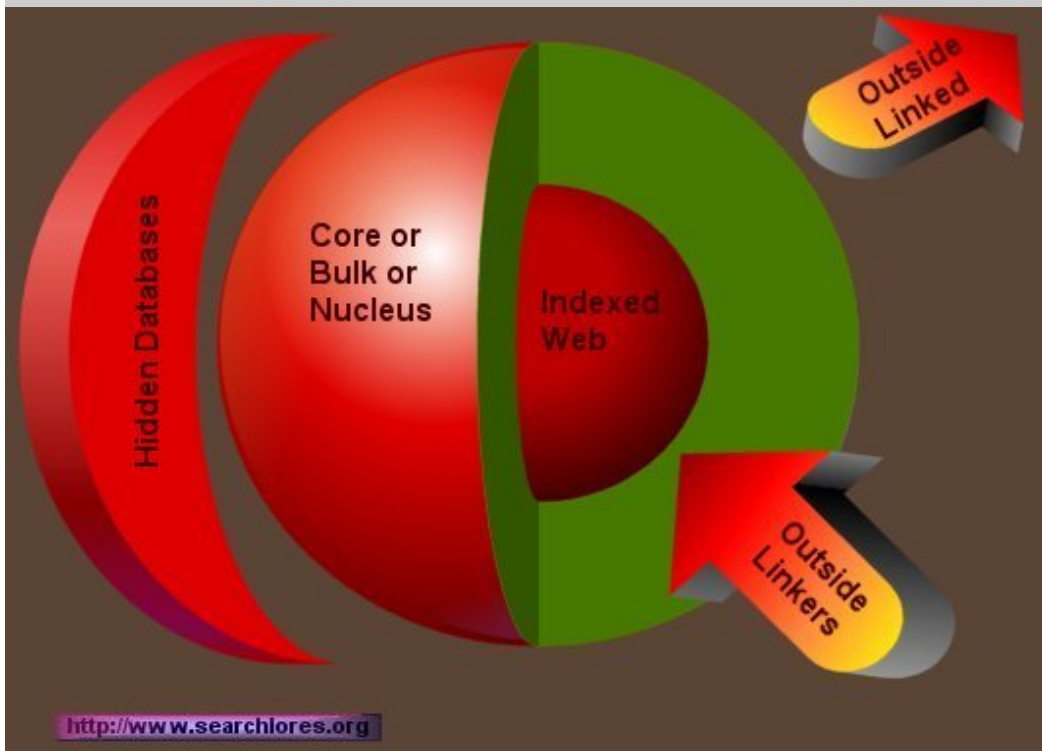
[search tips](#)

Example: site contains [\[searching\]](#) (a thousand sites eh!)

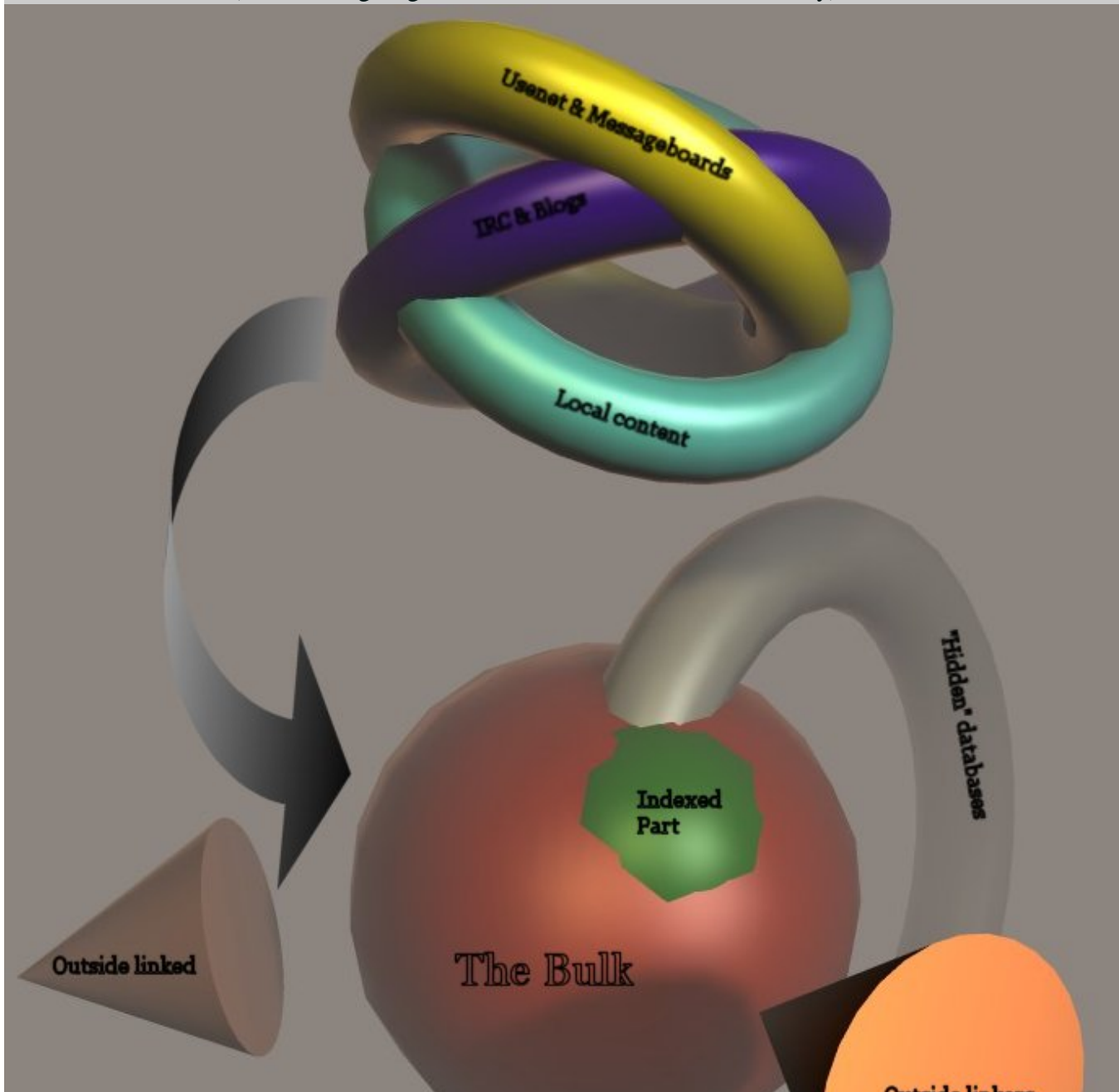
In fact Netcraft is so useful that you may want to add a [netcraft](#) javascript ad hoc bookmarklet to your bookmarks :-)

"SLIDES"

Structure of the web (the "classic" model)

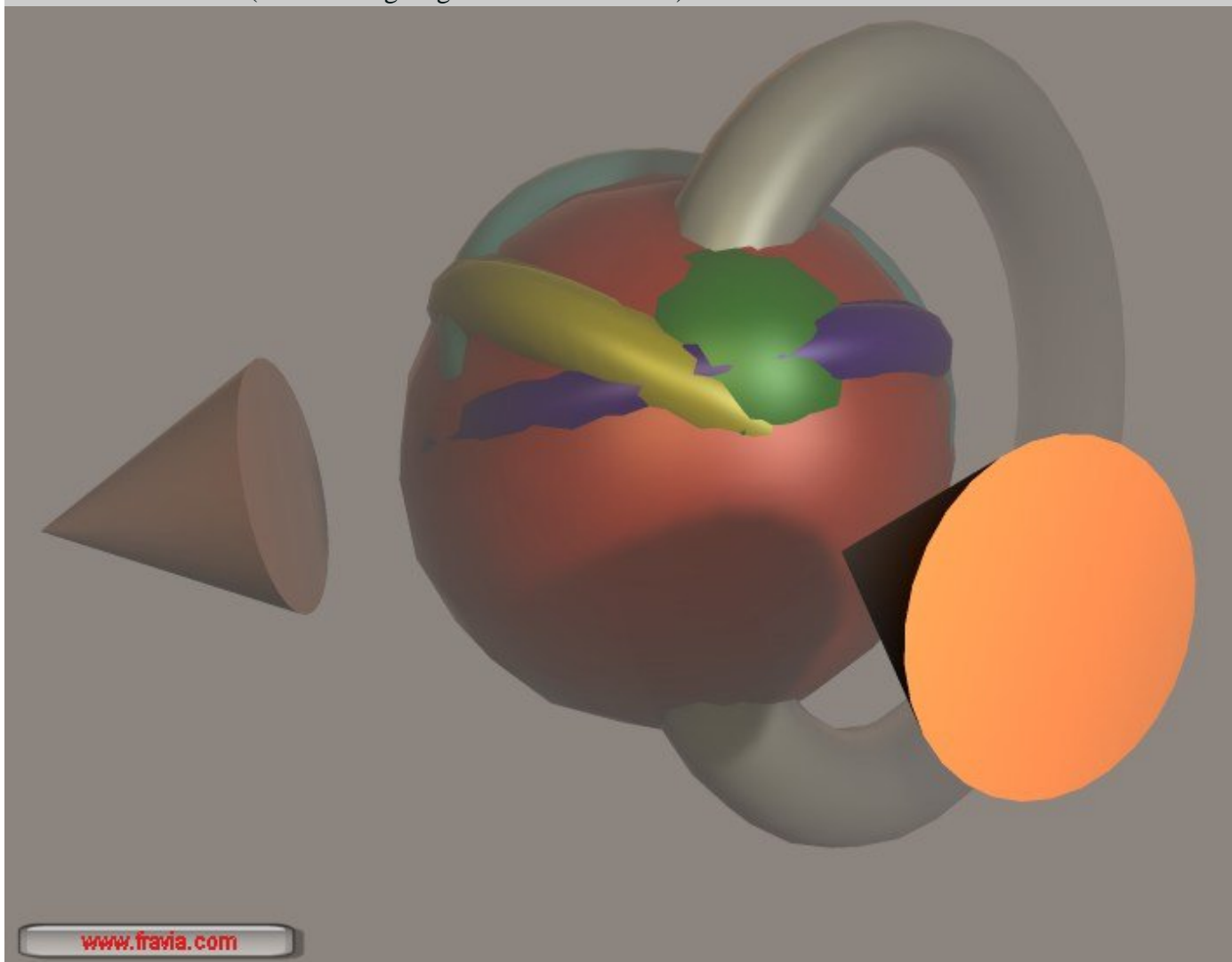


Structure of the web (the three big rings are shown outside the bulk for clarity)

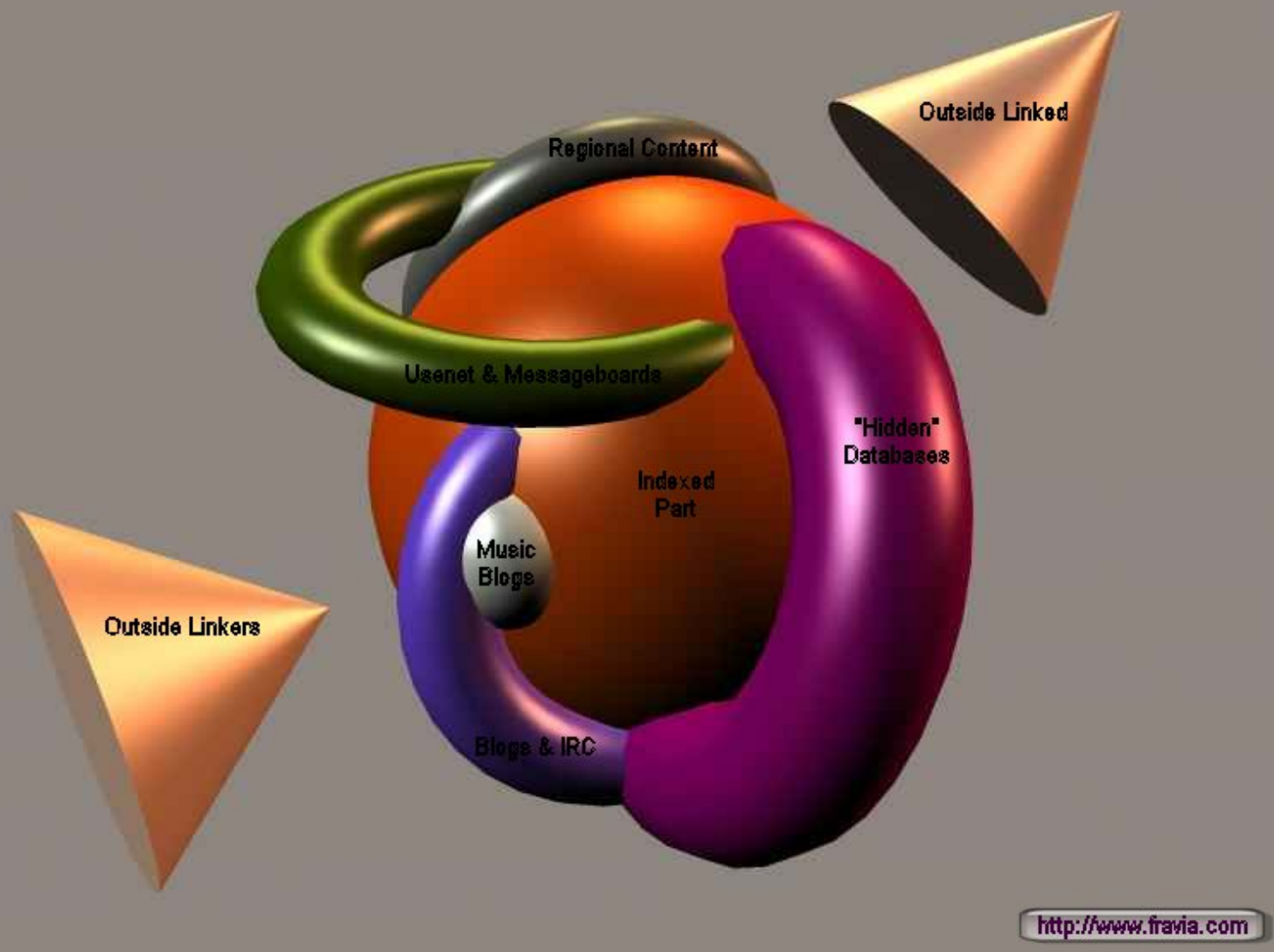




Structure of the web (the three big rings are inside the bulk)

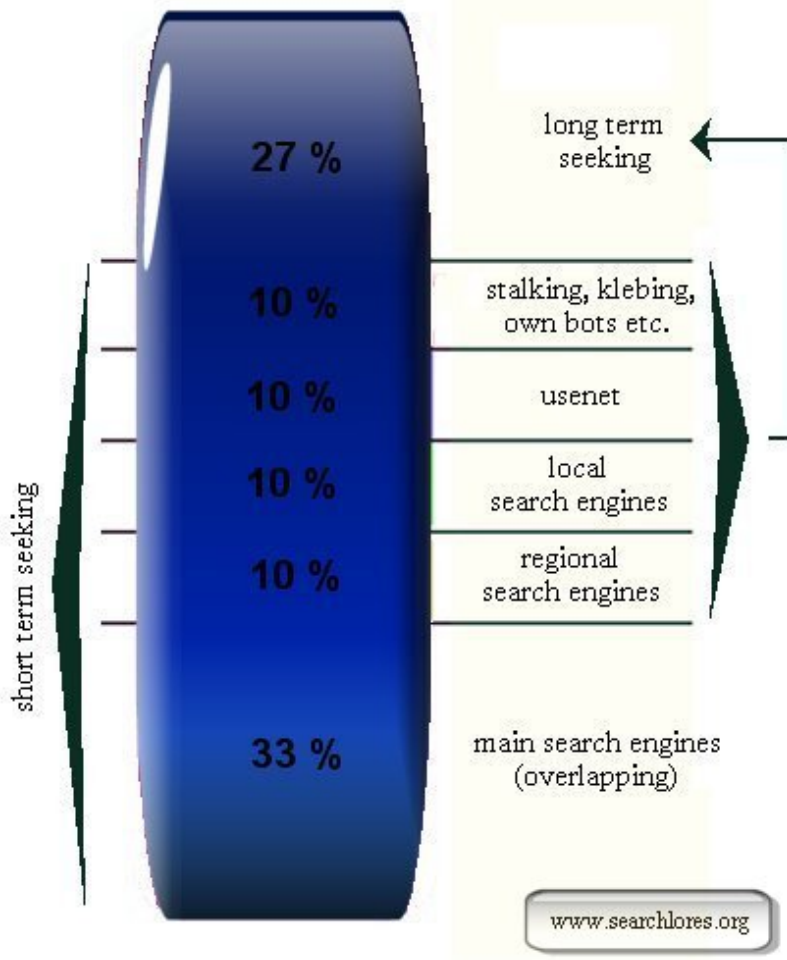


Structure of the web (a better image)



Short and long term seeking

Short / long term seeking total coverage of the Web (rough estimate)



LANGUAGES

Languages (that "english mothertongues" mostly underestimate): es: japanese bookmarklets... 🌐

As an example of how powerful some on-line services can be have for example a look at the following tool you lay use to understand a Japanese site,:

RIKAI

An incredible jappo-english translator!

<http://www.rikai.com/perl/Home.pl>

Try it for instance onto <http://www.shirofan.com/> See? It "massages" WWW pages and places "popup translations" from the EDICT database behind the Japanese text!

for instance

http://www.rikai.com/perl/LangMediator.En.pl?mediate_uri=http%3A%2F%2Fwww.shirofan.com%2F

See?

You can use this tool to "guess" the meaning of many a japanese page or -and especially- japanese search engine options, even if you do not know Japanese :-)

You can easily understand how, in this way, you can -with the proper tools- explore the wealth of results that the japanese, chinese, korean... you name them... search engines may (and probably will) give you.

Let's search for "[spanish search engines](#)"... see?

Let's now search for "[buscadores hispanos](#)"... see?

EVENTUALLY

I would also like to draw your attention towards the paramount importance of [names](#) on the web... 🚫

The ethical aspect... 🚫

An unjust society... 🚫

websearch importance nowadays recognized and obvious, you'll see tomorrow :-)... 🚫

libraries and documents: frills and substance... 🚫

the guardian of the light tower, the young kid in central africa and the yuppie in new york... 🚫

A (not so easy) "assignment" for reverse engineers turned seekers: find out what happened to Peter Urbanik, the author of wdasm...

Also remember that there are not only files on the web, but also solutions... the airport noise example.

CONCLUSIONS

Ode to the seekers

Like a skilled native, the able seeker has become [part of the web](#). He knows the smell of his forest: the foul-smelling mud of the popups, the slime of a rotting commercial javascript. He knows the sounds of the web: the gentle rustling of the [jpgs](#), the cries of the brightly colored [mp3s](#) that chase one another among the trees, singing as they go; the dark snuffling of the [m4as](#), the mechanical, monotone clinking of the huge, blind databases, the pathetic cry of the common user: a plaintive cooing that slides from one useless page down to the next until it dies away in a sad, little moan. In fact, to all those who do not understand it, today's Internet looks more and more like a closed, hostile and terribly boring commercial world.

Yet if you stop and hear attentively, you may be able to hear the seekers, deep into the shadows, singing a lusty chorus of praise to this wonderful world of theirs -- a world that gives them everything they want.

[The web is the habitat of the seeker, and in return for his knowledge and skill it satisfies all his needs.](#)

The seeker does not even need any more to hoard on his hard disks whatever he has found: all the various images, musics, films, books and whatnots that he fetches from the web... he can just [taste and leave there](#) what he finds, without even copying it, because he knows that nothing can disappear any more: once anything lands on the web, it will always be there, available for the eternity to all those that possess its secret [name](#)...

[The web-quicksand moves all the time, yet nothing can sink.](#)

In order to fetch all kinds of delicious fruits, the seeker just needs to raise his sharp searchstrings.

In perfect harmony with the surrounding internet forest, he can fetch again and again, at will, any target he fancies, wherever it may have been "hidden". The seeker moves unseen among sites and backbones, using his anonymity skills, his powerful proxomitron shield and his mighty HOST file.

If need be, he can quickly hide among the zombies, mimicking their behavior and thus disappearing into the mass.

Moving silently along the cornucopial forest of his web, picking his fruits and digging his jewels, the seeker avoids easily the many vicious traps that have been set to catch all the furry, sad little animals that happily use MSIE (and outlook), that use only one-word google "searches", and that browse and chat around all the time without proxies, bouncing against trackers and web-bugs and smearing all their personal data around.

Moreover the seeker is armed: his sharp browser will quickly cut to pieces any slimy javascript or rotting advertisement that the commercial beasts may have put on his way. His bots' jaws will tear apart any database defense, his powerful scripts will send perfectly balanced searchstrings far into the forest.

NOTES

Note_1)

Well, actually not so easy if you do not know how to enable it: open "[proxy](#)", rightclick on a proxy in the small window, choose advanced proxy setting, choose "[rotate proxy after every x connections](#)", or "[randomize rotations](#)".

Note_2)

Another interesting bookmarklet for password breaking purposes is the "word frequency" bookmarklet [word frequency](#)

Note_3)

Pestering php commercial vultures

In order to prepare your own "magic" tricks, and enter where you'r not supposed to (which is the "raison d'être" of the real seeker) you'll have to know the most common scripts used on the web.

This is quite rewarding, but you'll have to "dirty your hands" with some commercial fetid scripts. Knowing their putrid php code can be *quite* useful in order to reverse the helluja out of it when we encounter them in our wanderings.

Database scripts are -per definition- easy to retrieve, here a small list of the most easy to find on the web. They have been released long ago inside a package, so I feel free [to point to it](#), since anyone and his dog [already fetched and used them long ago](#) :-)

Auto Gallery SQL, AutoGallery Pro, Autolinks Pro, AutoRank Pro, Calendar Now Pro, ClickSee AdNow, DeskPRO Enterprise, Devil TGP, DigiShop, Done Right Bid Search Engine, e-Classifieds, ExplanationsSCripts.doc faqmaster.zip ImageFolio Commerce, Magic News Plus, Mojopersonals, Nendphp Publisher Enterprise, NewsPHP, Payment Gateway, Photopost Php Pro, PhotoPost, PHP Live Helper, PhpAuction PRO Plus, phpListPro, phpwebnews, pMachine, SmartSearch, Stardevelop Livehelp, SunShop, webDate, WebEdit Professional, X-affiliate, X-Cart.

Most of these scripts are in [php](#), and really easy to reverse for penetration purposes. Many more, different (and more clever) ones, are on the web alla round you. Go, fish, retrieve and multiply :-)

Note_4)

Music streaming à la facile

The simpliest way is just to take a cable and to plug the output back to the input socket.

Take a look at the back of your PC. Hopefully you are seeing the socket holes of your soundcard. One of them is surely an output (for your headphones), one of them is input (from a microphone?). Most times they both works with 3.5mm plugs. So if you have a cable with 3.5mm plugs in both sides you can loop back the music (or whatever output) to the soundcard/PC.

However, this is not that simple:

you got to carefully mute any other sound output apart the WAVE out, then in recording mute anything but the LINE IN.

then use a decent program like WAVELAB, and in wavelab set the rec level carefully to avoid noise or distortion. the quality will be inferior compared to recording the digital stream since in any case you record analog sound which was passed thru the AD/DA converter (which usually sucks unless you own a very costly audio card).

however, in the majority of the cases all you need is just to open WAVELAB and just record the WAVE OUT mix : you create a new track, press REC and select Wave Out as INPUT, you'll see the spectrogram moving etc, then when finished cut the recording and save it as wav or mp3 et voila'.

doing this you just record the digital stream and NOT the analog i/o, so the only downturn is that this stream is a frequency conversion but you'll hardly hear any difference from the original.

Note_5)

Crappabytes galore

Multiple ofbytes

Decimal prefixes			Binary prefixes		
Name	Symbol	Multiple	Name	Symbol	Multiple
kilobyte	kB	10 ³	kibibyte	KiB	2 ¹⁰
megabyte	MB	10 ⁶	mebibyte	MiB	2 ²⁰
gigabyte	GB	10 ⁹	gibibyte	GiB	2 ³⁰
terabyte	TB	10 ¹²	tebibyte	TiB	2 ⁴⁰
petabyte	PB	10 ¹⁵	pebibyte	PiB	2 ⁵⁰
exabyte	EB	10 ¹⁸	exbibyte	EiB	2 ⁶⁰
zettabyte	ZB	10 ²¹			
yottabyte	YB	10 ²⁴			



The Door



the Hall



The Library



The Studio



The Garden path

